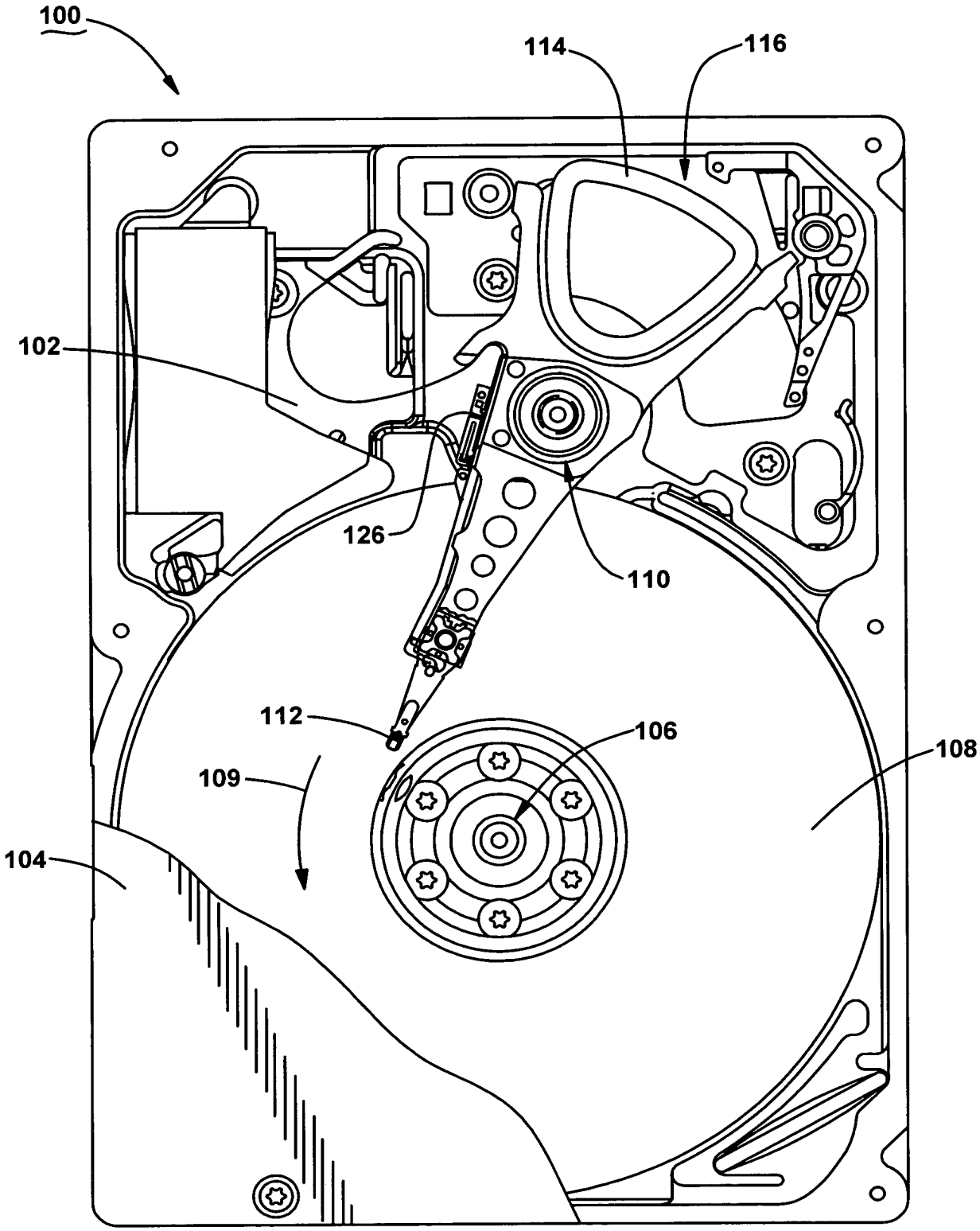

FIG. 1

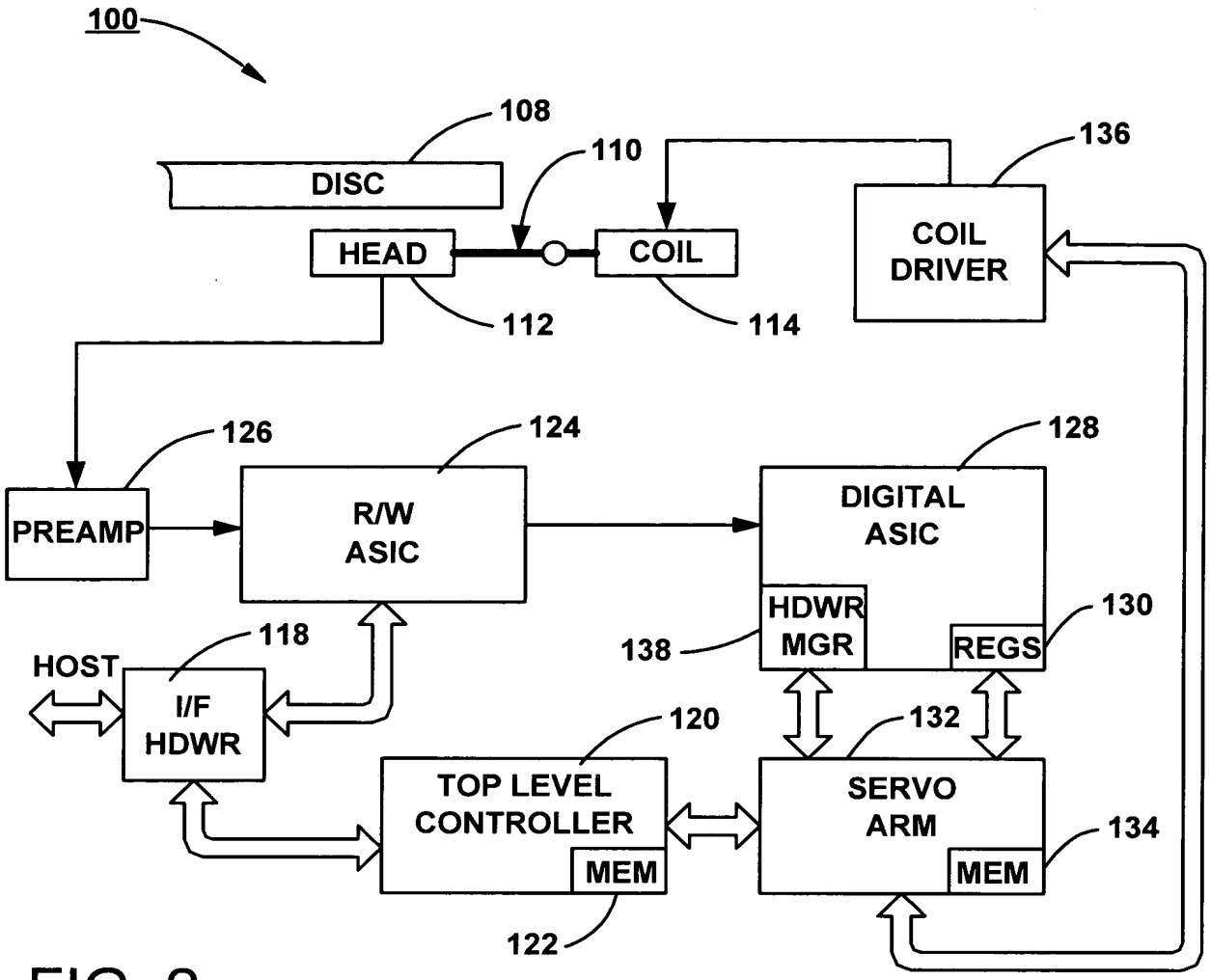
100
108
DISC
110
HEAD
COIL
112
114
136
COIL DRIVER
126
PREAMP
124
R/W ASIC
128
DIGITAL ASIC
HDWR MGR
138
REGS
130
HOST
118
I/F HDWR
120
132
TOP LEVEL CONTROLLER
MEM
SERVO ARM
MEM
134
122

FIG. 2

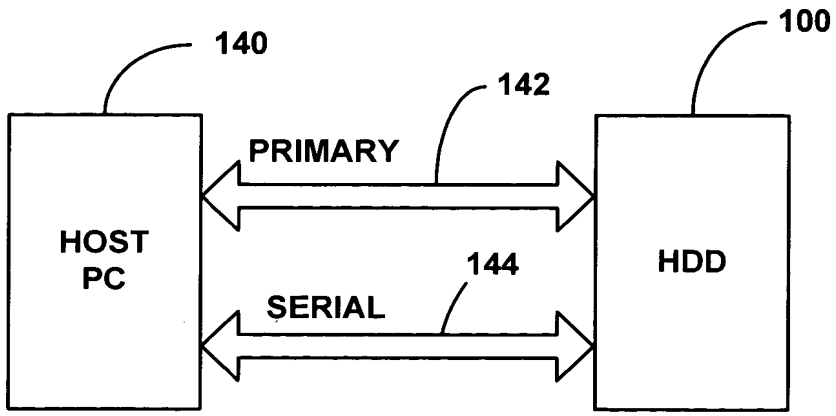
140
142
100
PRIMARY
HOST PC
HDD
144
SERIAL

FIG. 3

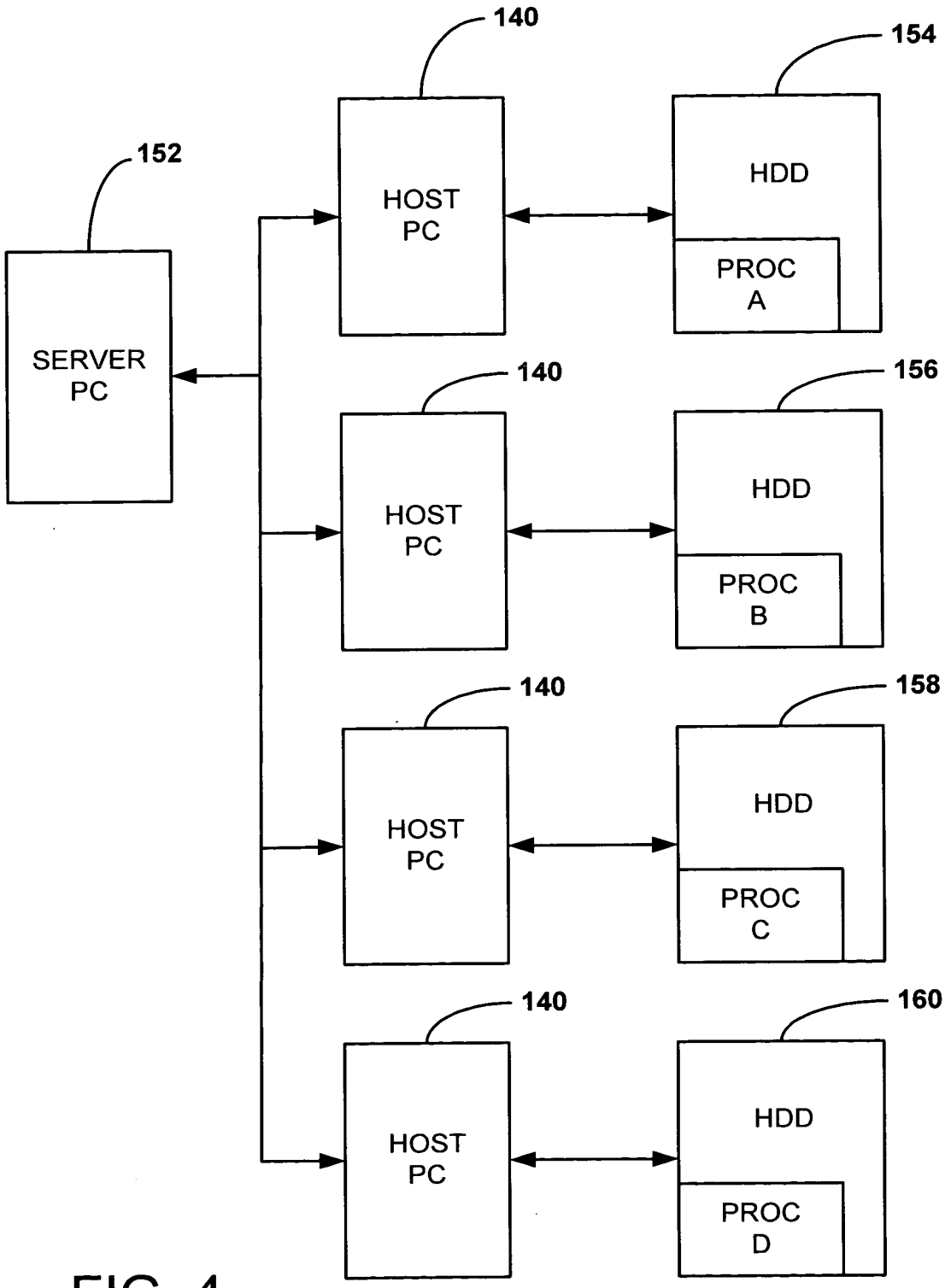
140
154
152
HOST PC
HDD
PROC A
SERVER PC
140
156
HOST PC
HDD
PROC B
140
158
HOST PC
HDD
PROC C
140
160
HOST PC
HDD
PROC D

FIG. 4

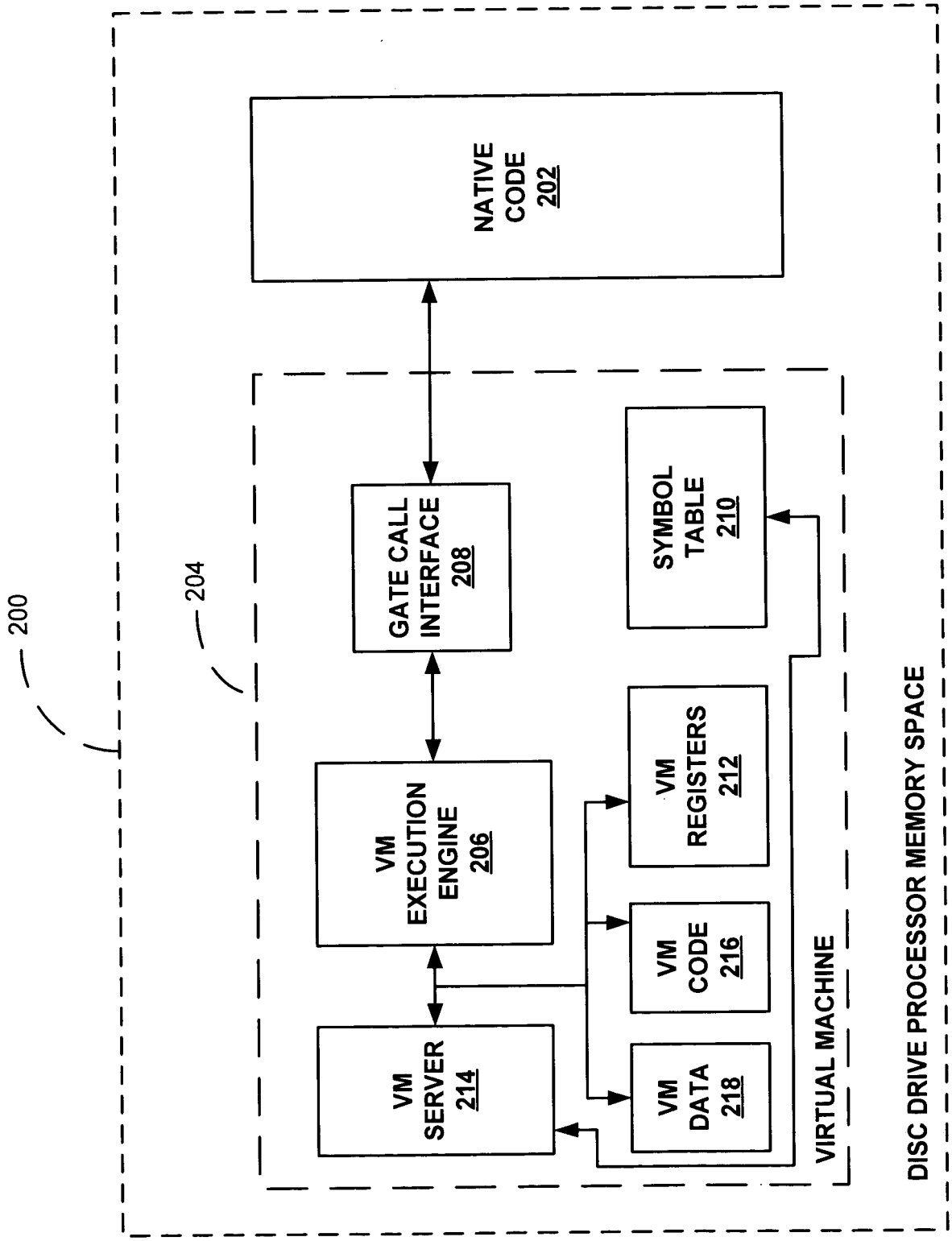
200
204
NATIVE CODE 202
GATE CALL INTERFACE 208
SYMBOL TABLE 210
VM EXECUTION ENGINE 206
VM REGISTERS 212
VM CODE 216
VM SERVER 214
VM DATA 218
VIRTUAL MACHINE
DISC DRIVE PROCESSOR MEMORY SPACE

FIG. 5

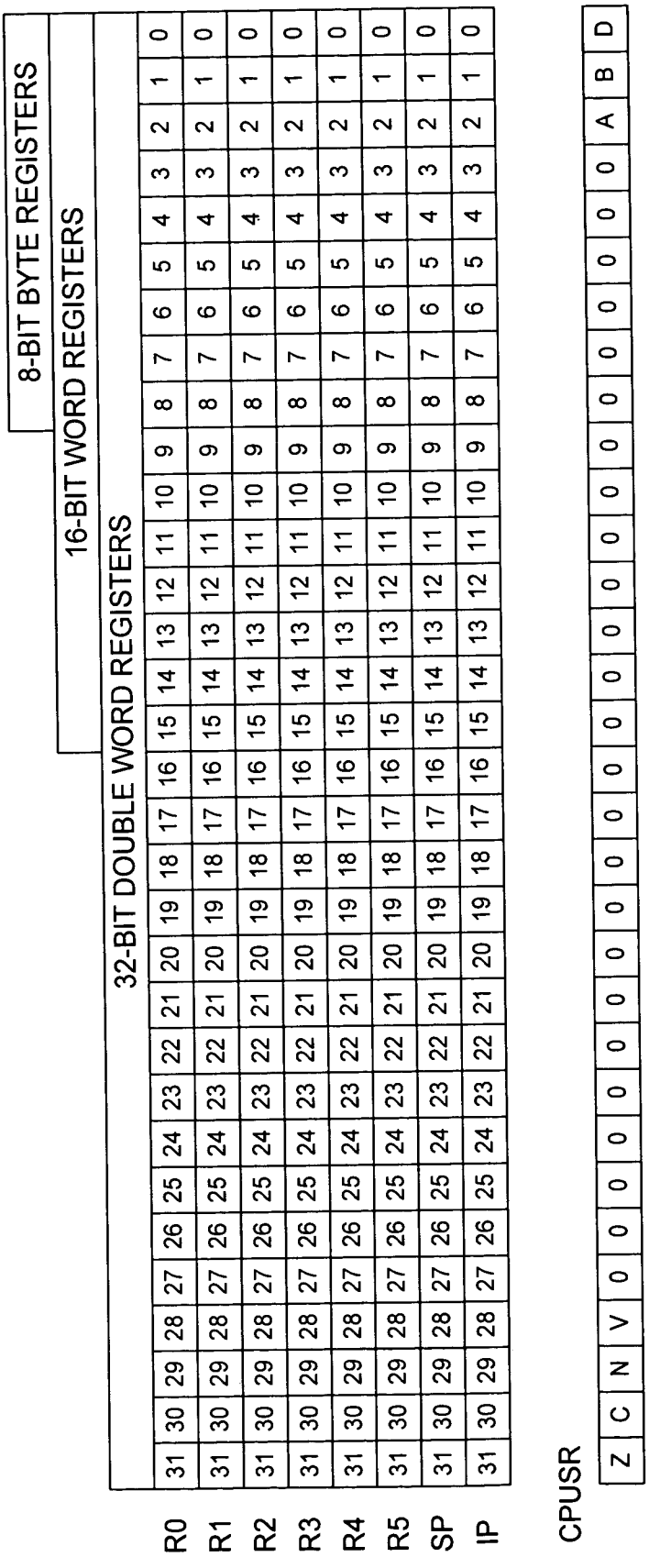
8-BIT BYTE REGISTERS
16-BIT WORD REGISTERS
32-BIT DOUBLE WORD REGISTERS
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
R0
R1
R2
R3
R4
R5
SP
IP
CPUSR
Z C N V 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A B D

FIG. 6

NAME	PARAMETERS	DESCRIPTION	Z	C	N	V	NOTES	BITS
LOAD/STORE/NOP								
NOP		NO OPERATION						0 0 1 0 0 0 0 0 X X X X X X X X
LDI	Rd, #VALUE	LOAD IMMEDIATE VALUE TO REGISTER						0 0 1 0 0 0 0 1 G G Rd X X X
LDG	Rd, Rp OR #ABSOLUTE	LOAD GLOBAL VARIABLE FROM MEMORY						0 0 1 0 0 0 1 1 G G Rd Rp
STG	Rd, Rp OR #ABSOLUTE	STORE GLOBAL VARIABLE TO MEMORY						0 0 1 0 0 1 0 1 G G Rs Rp
LDL	Rd, #OFFSET	LOAD LOCAL VARIABLE FROM MEMORY					ADDR=SP+#OFFSET	0 0 1 0 0 1 1 1 G G Rd X X X
STL	Rd, #OFFSET	STORE LOCAL VARIABLE TO MEMORY					ADDR=SP+#OFFSET	0 0 1 0 1 0 0 1 G G Rs X X X
SXX	Rs	SIGN EXTEND REGISTER TO 32 BIT						0 0 1 0 1 0 1 1 G G Rs X X X
MOV	Rd, Rs	REGISTER TO REGISTER MOVE						0 0 1 0 1 1 0 0 G G Rd Rs
TST	Rs	TEST REGISTER AND SET FLAGS	X		X			0 0 1 0 1 1 1 0 G G Rs X X X
ARITHMETIC								
ADD	Rd, Rv OR #VALUE	ADD	X	X	X	X		0 1 0 0 0 0 0 1 G G Rd Rv or X
ADC	Rd, Rv OR #VALUE	ADD WITH CARRY	X	X	X	X		0 1 0 0 0 0 1 1 G G Rd Rv or X
SUB	Rd, Rv OR #VALUE	SUBTRACT	X	X	X	X		0 1 0 0 0 1 0 1 G G Rd Rv or X
SBC	Rd, Rv OR #VALUE	SUBTRACT WITH CARRY	X	X	X	X		0 1 0 0 0 1 1 1 G G Rd Rv or X
CMP	Ra, Rv OR #VALUE	COMPARE	X	X	X	X		0 1 0 0 1 0 0 1 G G Ra Rv or X
ABS	Rd	ABSOLUTE VALUE						0 1 0 0 1 0 1 0 G G Rd X X X
MUL	Rd, Rv OR #VALUE	MULTIPLY				X	Rd=Rd*Rv (UNSIGNED)	0 1 0 0 1 1 0 1 G G Rd Rv or X
IMUL	Rd, Rv OR #VALUE	SIGNED MULTIPLY				X	Rd=Rd*Rv (SIGNED)	0 1 0 0 1 1 1 1 G G Rd Rv or X
DIV	Rd, Rv OR #VALUE	DIVIDE					Rd=Rd/Rv (UNSIGNED)	0 1 0 1 0 0 0 1 G G Rd Rv or X
IDIV	Rd, Rv OR #VALUE	SIGNED DIVIDE					Rd=Rd/Rv (SIGNED)	0 1 0 1 0 0 1 1 G G Rd Rv or X
MOD	Rd, Rv OR #VALUE	MODULUS					Rd=Rd%Rv (UNSIGNED)	0 1 0 1 0 1 0 1 G G Rd Rv or X
IMOD	Rd, Rv OR #VALUE	SIGNED MODULUS					Rd=Rd%*Rv (SIGNED)	0 1 0 1 0 1 1 1 G G Rd Rv or X
LOGICAL								
AND	Rd, Rm OR #MASK	BITWISE LOGICAL AND						0 1 1 0 0 0 0 1 G G Rd Rm OR X
ORR	Rd, Rm OR #MASK	BITWISE LOGICAL OR						0 1 1 0 0 0 1 1 G G Rd Rm OR X
XOR	Rd, Rm OR #MASK	BITWISE LOGICAL XOR						0 1 1 0 0 1 0 1 G G Rd Rm OR X
NOT	Rd	BITWISE COMPLEMENT						0 1 1 0 0 1 1 0 G G Rd X X X

FIG. 7

NAME	PARAMETERS	DESCRIPTION	Z	C	N	V	NOTES	BITS
SHIFT								
LSL	Rd, Rn	LOGICAL SHIFT LEFT					SHIFT LEFT, ZERO FILL	1 0 0 0 0 0 0 0 G G Rd Rn
LSR	Rd, Rn	LOGICAL SHIFT RIGHT					SHIFT RT, ZERO FILL	1 0 0 0 0 0 1 0 G G Rd Rn
ASR	Rd, Rn	ARITHMETIC SHIFT RIGHT					SHIFT RT, SIGNBIT FILL	1 0 0 0 0 1 0 0 G G Rd Rn
ROR	Rd, Rn	ROTATE RIGHT		X			ROT RT INTO CARRY	1 0 0 0 0 1 1 0 G G Rd Rn
RRX	Rd, Rn	ROTATE RIGHT EXTENDED		X			ROT RT THRU CARRY	1 0 0 0 1 0 0 0 G G Rd Rn
CLC		CLEAR CARRY		X				1 0 0 0 1 0 1 0 X X X X X X X X
STC		SET CARRY		X				1 0 0 0 1 1 0 0 X X X X X X X X
CMC		COMPLEMENT CARRY		X				1 0 0 0 1 1 1 0 X X X X X X X X
BRANCH								
JMP	#OFFSET	JUMP					ALWAYS	1 0 1 0 0 0 0 1 0 0 X X X X X X
JEQ	#OFFSET	JUMP EQUAL					Z SET	1 0 1 0 0 0 1 1 0 0 X X X X X X
JNE	#OFFSET	JUMP NOT EQUAL					Z CLEAR	1 0 1 0 0 1 0 1 0 0 X X X X X X
JPL	#OFFSET	JUMP PLUS					N CLEAR	1 0 1 0 0 1 1 1 0 0 X X X X X X
JMI	#OFFSET	JUMP MINUS					N SET	1 0 1 0 1 0 0 1 0 0 X X X X X X
JLO	#OFFSET	JUMP LOWER (UNSIGNED COMPARE)					C CLEAR	1 0 1 0 1 0 1 1 0 0 X X X X X X
JHS	#OFFSET	JUMP HIGHER SAME (UNSIGNED COMPARE)					C SET	1 0 1 0 1 1 0 1 0 0 X X X X X X
JVC	#OFFSET	JUMP OVERFLOW CLEAR					V CLEAR	1 0 1 0 1 1 1 1 0 0 X X X X X X
JVS	#OFFSET	JUMP OVERFLOW SET					VSET	1 0 1 1 0 0 0 1 0 0 X X X X X X
JGT	#OFFSET	JUMP GREATER THAN (SIGNED COMPARE)					Z CLEAR AND N=V	1 0 1 1 0 0 1 1 0 0 X X X X X X
JGE	#OFFSET	JUMP GREATER EQUAL (SIGNED COMPARE)					N=V	1 0 1 1 0 1 0 1 0 0 X X X X X X
JLT	#OFFSET	JUMP LESS THAN (SIGNED COMPARE)					N<>V	1 0 1 1 0 1 1 1 0 0 X X X X X X
JLE	#OFFSET	JUMP LESS EQUAL (SIGNED COMPARE)					Z SET OR N<>V	1 0 1 1 1 0 0 1 0 0 X X X X X X
JHI	#OFFSET	JUMP HIGHER (UNSIGNED COMPARE)					C SET AND Z CLEAR	1 0 1 1 1 0 1 1 0 0 X X X X X X
JLS	#OFFSET	JUMP LESS SAME (UNSIGNED COMPARE)					C CLEAR OR Z SET	1 0 1 1 1 1 0 1 0 0 X X X X X X
CALL	Rp OR #ABSOLUTE(32)	CALL SUBROUTINE					PUSH PC, CALL	1 0 1 1 1 1 1 1 0 0 Rp X X X
STACK								
PUSH	Rs	PUSH Rs ONTO STACK					SP=SP-SIZEOF(Rs),*SP=Rs	1 1 0 0 0 0 0 0 G G Rs X X X
POP	Rd	POP Rd FROM STACK					RD=*SP,SP=SP+SIZEOF(Rd)	1 1 0 0 0 0 1 0 G G Rd X X X
GATE CALL								
GCALL	Rg	Rg IS ADDRESS VALUE IN SYMBOL TABLE						1 1 1 0 0 0 0 0 0 0 Rg X X X

FIG. 8

Title: VIRTUAL MACHINE EMULATION IN THE MEMORY..
Inventors: Chad R. Overton, et al. .
Dkt. No.: 72940/02-364
Filed: October 3, 2003

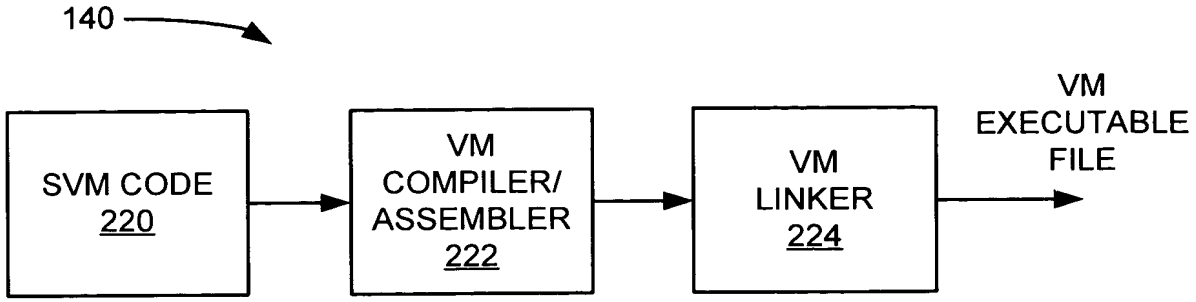
140
SVM CODE
220
VM
COMPILER/
ASSEMBLER
222
VM
LINKER
224
VM
EXECUTABLE
FILE

FIG. 9

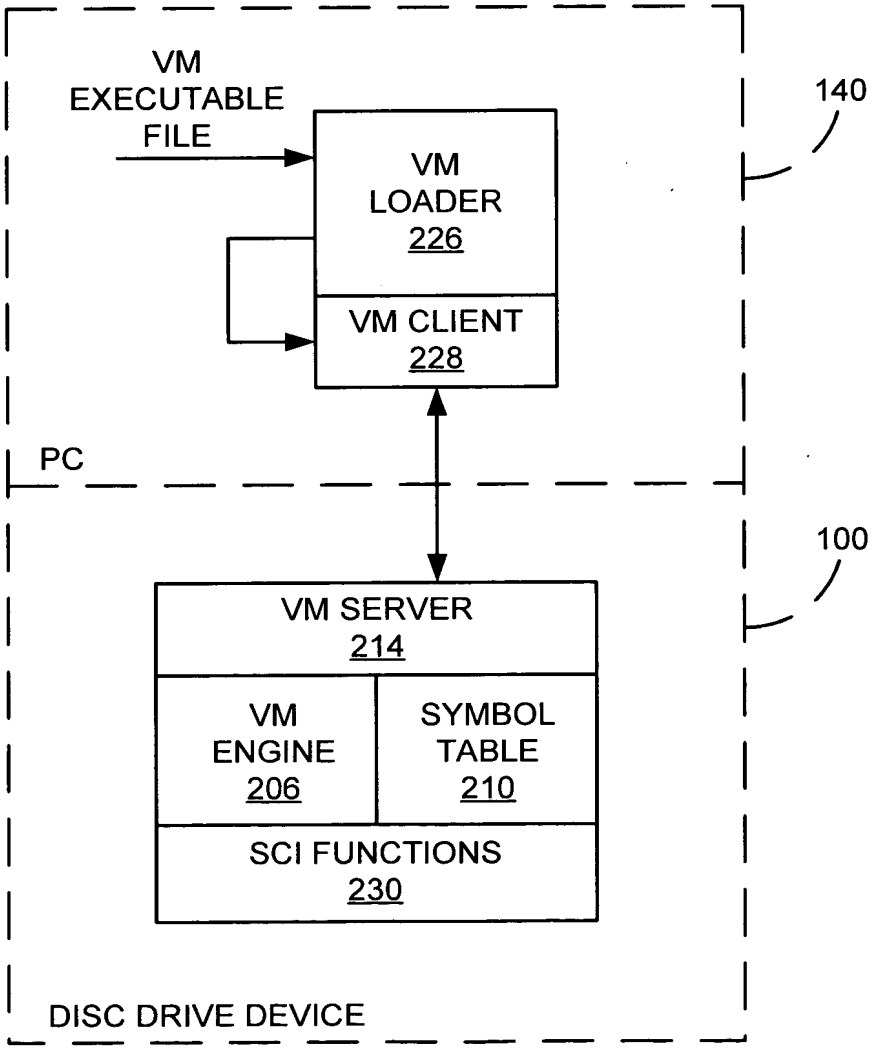
VM
EXECUTABLE
FILE
VM
LOADER
226
VM CLIENT
228
140
PC
100
VM SERVER
214
VM
ENGINE
206
SYMBOL
TABLE
210
SCI FUNCTIONS
230
DISC DRIVE DEVICE

FIG. 10

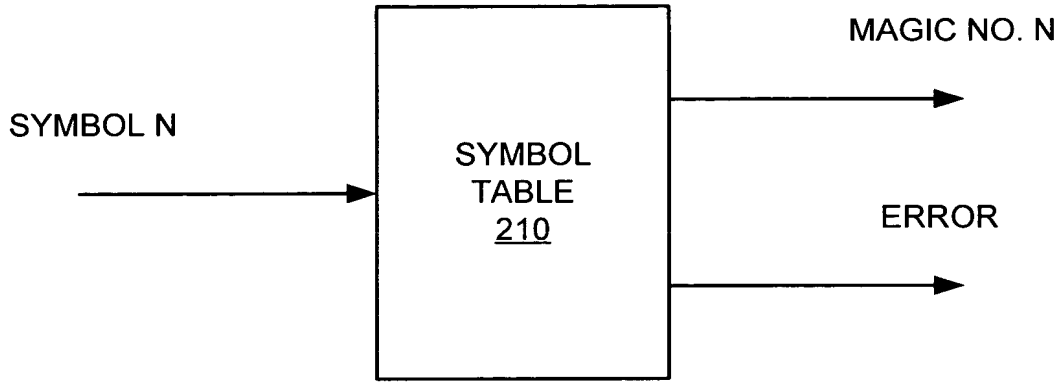
SYMBOL N
SYMBOL
TABLE
210
MAGIC NO. N
ERROR

FIG. 11

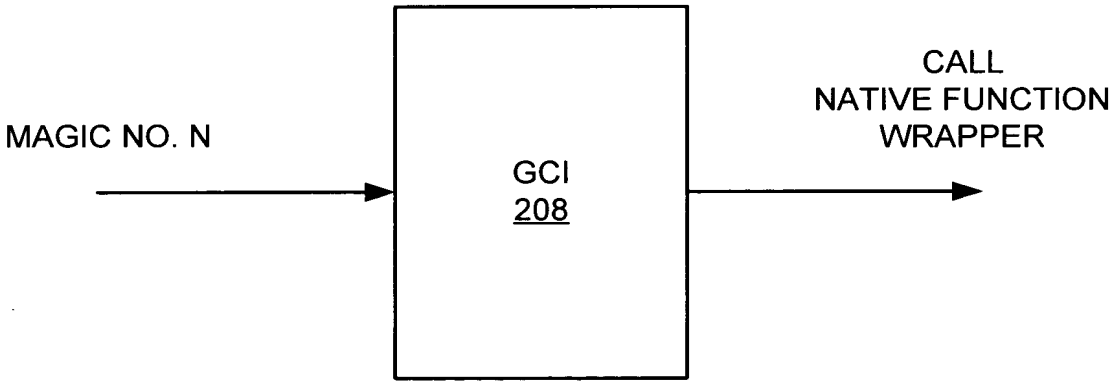
MAGIC NO. N
GCI
208
CALL
NATIVE FUNCTION
WRAPPER

FIG. 12

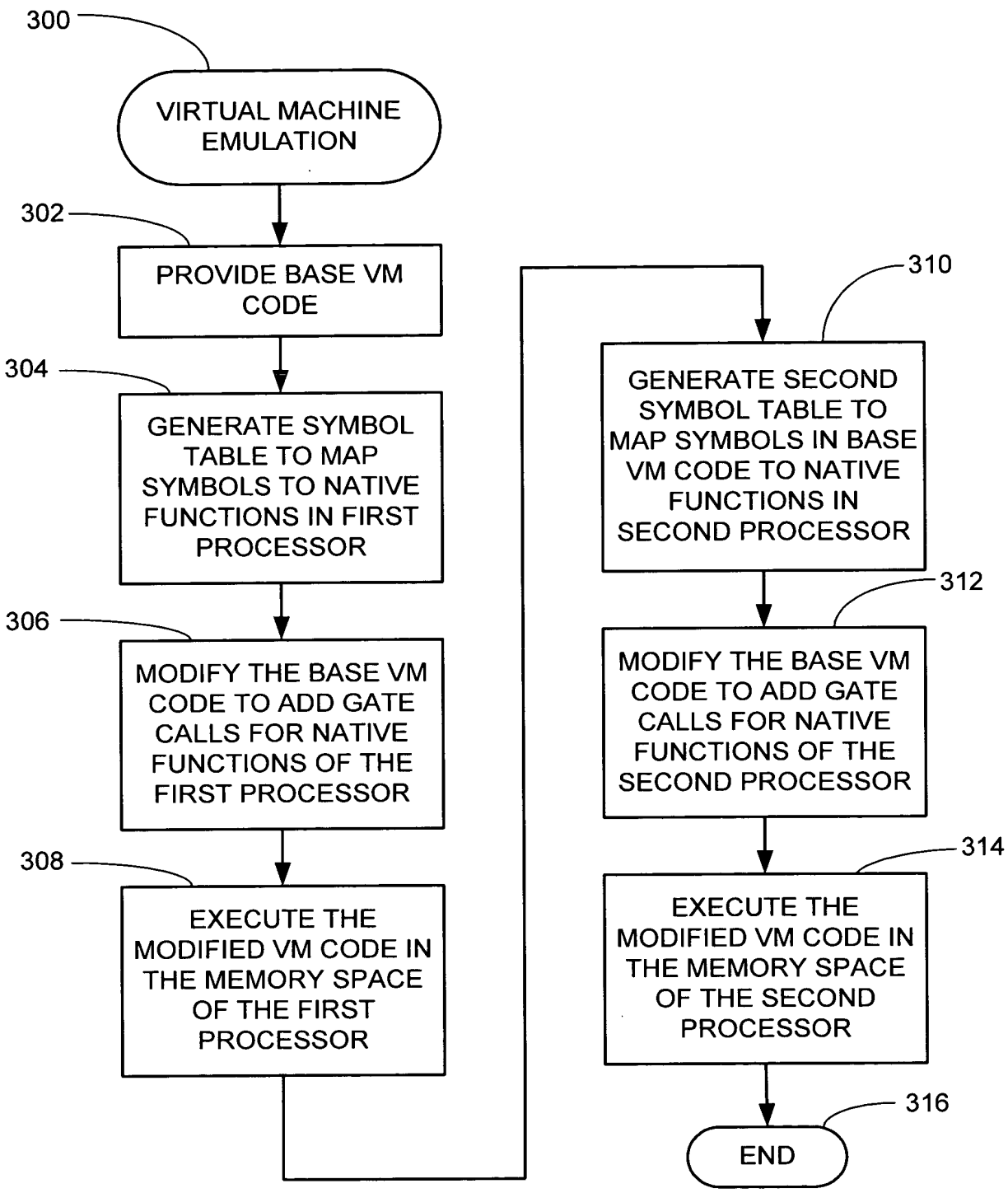
300
VIRTUAL MACHINE EMULATION
302
PROVIDE BASE VM CODE
304
GENERATE SYMBOL TABLE TO MAP SYMBOLS TO NATIVE FUNCTIONS IN FIRST PROCESSOR
306
MODIFY THE BASE VM CODE TO ADD GATE CALLS FOR NATIVE FUNCTIONS OF THE FIRST PROCESSOR
308
EXECUTE THE MODIFIED VM CODE IN THE MEMORY SPACE OF THE FIRST PROCESSOR
310
GENERATE SECOND SYMBOL TABLE TO MAP SYMBOLS IN BASE VM CODE TO NATIVE FUNCTIONS IN SECOND PROCESSOR
312
MODIFY THE BASE VM CODE TO ADD GATE CALLS FOR NATIVE FUNCTIONS OF THE SECOND PROCESSOR
314
EXECUTE THE MODIFIED VM CODE IN THE MEMORY SPACE OF THE SECOND PROCESSOR
316
END

FIG. 13